Jon Postel

Network Working Group
Request for Comments: 1101
Updates: RFCs 1034, 1035

P. Mockapetris
ISI
April 1989

DNS Encoding of Network Names and Other Types

1. STATUS OF THIS MEMO

This RFC proposes two extensions to the Domain Name System:

- A specific method for entering and retrieving RRs which map between network names and numbers.

- Ideas for a general method for describing mappings between arbitrary identifiers and numbers.

The method for mapping between network names and addresses is a proposed standard, the ideas for a general method are experimental.

This RFC assumes that the reader is familiar with the DNS [RFC 1034, RFC 1035] and its use. The data shown is for pedagogical use and does not necessarily reflect the real Internet.

Distribution of this memo is unlimited.

2. INTRODUCTION

The DNS is extensible and can be used for a virtually unlimited number of data types, name spaces, etc. New type definitions are occasionally necessary as are revisions or deletions of old types (e.g., MX replacement of MD and MF [RFC 974]), and changes described in [RFC 973]. This RFC describes changes due to the general need to map between identifiers and values, and a specific need for network name support.

Users wish to be able to use the DNS to map between network names and numbers. This need is the only capability found in HOSTS.TXT which is not available from the DNS. In designing a method to do this, there were two major areas of concern:

- Several tradeoffs involving control of network names, the syntax of network names, backward compatibility, etc.

- A desire to create a method which would be sufficiently general to set a good precedent for future mappings, for example, between TCP-port names and numbers,

autonomous system names and numbers, X.500 Relative Distinguished Names (RDNs) and their servers, or whatever.

It was impossible to reconcile these two areas of concern for network names because of the desire to unify network number support within existing IP address to host name support. The existing support is the IN-ADDR.ARPA section of the DNS name space. As a result this RFC describes one structure for network names which builds on the existing support for host names, and another family of structures for future yellow pages (YP) functions such as conversions between TCP-port numbers and mnemonics.

Both structures are described in following sections. Each structure has a discussion of design issues and specific structure recommendations.

We wish to avoid defining structures and methods which can work but do not because of indifference or errors on the part of system administrators when maintaining the database. The WKS RR is an example. Thus, while we favor distribution as a general method, we also recognize that centrally maintained tables (such as HOSTS.TXT) are usually more consistent though less maintainable and timely. Hence we recommend both specific methods for mapping network names, addresses, and subnets, as well as an instance of the general method for mapping between allocated network numbers and network names. (Allocation is centrally performed by the SRI Network Information Center, aka the NIC).

3. NETWORK NAME ISSUES AND DISCUSSION

The issues involved in the design were the definition of network name syntax, the mappings to be provided, and possible support for similar functions at the subnet level.

3.1. Network name syntax

The current syntax for network names, as defined by [RFC 952] is an alphanumeric string of up to 24 characters, which begins with an alpha, and may include "." and "-" except as first and last characters. This is the format which was also used for host names before the DNS. Upward compatibility with existing names might be a goal of any new scheme.

However, the present syntax has been used to define a flat name space, and hence would prohibit the same distributed name allocation method used for host names. There is some sentiment for allowing the NIC to continue to allocate and regulate network names, much as it allocates numbers, but the majority opinion favors local control of

network names. Although it would be possible to provide a flat space or a name space in which, for example, the last label of a domain name captured the old-style network name, any such approach would add complexity to the method and create different rules for network names and host names.

For these reasons, we assume that the syntax of network names will be the same as the expanded syntax for host names permitted in [HR]. The new syntax expands the set of names to allow leading digits, so long as the resulting representations do not conflict with IP addresses in decimal octet form. For example, 3Com.COM and 3M.COM are now legal, although 26.0.0.73.COM is not. See [HR] for details.

The price is that network names will get as complicated as host names. An administrator will be able to create network names in any domain under his control, and also create network number to name entries in IN-ADDR.ARPA domains under his control. Thus, the name for the ARPANET might become NET.ARPA, ARPANET.ARPA or Arpa-network.MIL., depending on the preferences of the owner.

3.2. Mappings

The desired mappings, ranked by priority with most important first, are:

- Mapping a IP address or network number to a network name.

  This mapping is for use in debugging tools and status displays of various sorts. The conversion from IP address to network number is well known for class A, B, and C IP addresses, and involves a simple mask operation. The needs of other classes are not yet defined and are ignored for the rest of this RFC.

- Mapping a network name to a network address.

  This facility is of less obvious application, but a symmetrical mapping seems desirable.

- Mapping an organization to its network names and numbers.

  This facility is useful because it may not always be possible to guess the local choice for network names, but the organization name is often well known.

- Similar mappings for subnets, even when nested.

  The primary application is to be able to identify all of the subnets involved in a particular IP address. A secondary

requirement is to retrieve address mask information.

3.3. Network address section of the name space

The network name syntax discussed above can provide domain names which will contain mappings from network names to various quantities, but we also need a section of the name space, organized by network and subnet number to hold the inverse mappings.

The choices include:

- The same network number slots already assigned and delegated in the IN-ADDR.ARPA section of the name space.

  For example, 10.IN-ADDR.ARPA for class A net 10, 2.128.IN-ADDR.ARPA for class B net 128.2, etc.

- Host-zero addresses in the IN-ADDR.ARPA tree. (A host field of all zero in an IP address is prohibited because of confusion related to broadcast addresses, et al.)

  For example, 0.0.0.10.IN-ADDR.ARPA for class A net 10, 0.0.2.128.IN-ADDR.arpa for class B net 128.2, etc. Like the first scheme, it uses in-place name space delegations to distribute control.

  The main advantage of this scheme over the first is that it allows convenient names for subnets as well as networks. A secondary advantage is that it uses names which are not in use already, and hence it is possible to test whether an organization has entered this information in its domain database.

- Some new section of the name space.

  While this option provides the most opportunities, it creates a need to delegate a whole new name space. Since the IP address space is so closely related to the network number space, most believe that the overhead of creating such a new space is overwhelming and would lead to the WKS syndrome. (As of February, 1989, approximately 400 sections of the IN-ADDR.ARPA tree are already delegated, usually at network boundaries.)

4. SPECIFICS FOR NETWORK NAME MAPPINGS

The proposed solution uses information stored at:

- Names in the IN-ADDR.ARPA tree that correspond to host-zero IP addresses. The same method is used for subnets in a nested fashion. For example, 0.0.0.10.IN-ADDR.ARPA. for net 10.

  Two types of information are stored here: PTR RRs which point to the network name in their data sections, and A RRs, which are present if the network (or subnet) is subnetted further. If a type A RR is present, then it has the address mask as its data. The general form is:

  ```
  <reversed-host-zero-number>.IN-ADDR.ARPA. PTR <network-name>
  <reversed-host-zero-number>.IN-ADDR.ARPA. A   <subnet-mask>
  ```

  For example:

  ```
  0.0.0.10.IN-ADDR.ARPA.   PTR     ARPANET.ARPA.
  ```

  or

  ```
  0.0.2.128.IN-ADDR.ARPA. PTR     cmu-net.cmu.edu.
                          A       255.255.255.0
  ```

  In general, this information will be added to an existing master file for some IN-ADDR.ARPA domain for each network involved. Similar RRs can be used at host-zero subnet entries.

- Names which are network names.

  The data stored here is PTR RRs pointing at the host-zero entries. The general form is:

  ```
  <network-name> ptr <reversed-host-zero-number>.IN-ADDR.ARPA
  ```

  For example:

  ```
  ARPANET.ARPA.              PTR     0.0.0.10.IN-ADDR.ARPA.
  ```

  or

  ```
  isi-net.isi.edu.           PTR     0.0.9.128.IN-ADDR.ARPA.
  ```

  In general, this information will be inserted in the master file for the domain name of the organization; this is a

different file from that which holds the information below IN-ADDR.ARPA. Similar PTR RRs can be used at subnet names.

- Names corresponding to organizations.

  The data here is one or more PTR RRs pointing at the IN-ADDR.ARPA names corresponding to host-zero entries for networks.

  For example:

```
ISI.EDU.          PTR     0.0.9.128.IN-ADDR.ARPA.

MCC.COM.          PTR     0.167.5.192.IN-ADDR.ARPA.
                  PTR     0.168.5.192.IN-ADDR.ARPA.
                  PTR     0.169.5.192.IN-ADDR.ARPA.
                  PTR     0.0.62.128.IN-ADDR.ARPA.
```

4.1. A simple example

The ARPANET is a Class A network without subnets. The RRs which would be added, assuming the ARPANET.ARPA was selected as a network name, would be:

```
ARPA.                     PTR     0.0.0.10.IN-ADDR.ARPA.

ARPANET.ARPA.             PTR     0.0.0.10.IN-ADDR.ARPA.

0.0.0.10.IN-ADDR.ARPA.    PTR     ARPANET.ARPA.
```

The first RR states that the organization named ARPA owns net 10 (It might also own more network numbers, and these would be represented with an additional RR per net.) The second states that the network name ARPANET.ARPA. maps to net 10. The last states that net 10 is named ARPANET.ARPA.

Note that all of the usual host and corresponding IN-ADDR.ARPA entries would still be required.

4.2. A complicated, subnetted example

The ISI network is 128.9, a class B number. Suppose the ISI network was organized into two levels of subnet, with the first level using an additional 8 bits of address, and the second level using 4 bits, for address masks of x'FFFFFF00' and X'FFFFFFF0'.

Then the following RRs would be entered in ISI's master file for the ISI.EDU zone:

```
; Define network entry
isi-net.isi.edu.                PTR  0.0.9.128.IN-ADDR.ARPA.

; Define first level subnets
div1-subnet.isi.edu.            PTR  0.1.9.128.IN-ADDR.ARPA.
div2-subnet.isi.edu.            PTR  0.2.9.128.IN-ADDR.ARPA.

; Define second level subnets
inc-subsubnet.isi.edu.          PTR  16.2.9.128.IN-ADDR.ARPA.
```

in the 9.128.IN-ADDR.ARPA zone:

```
; Define network number and address mask
0.0.9.128.IN-ADDR.ARPA.         PTR  isi-net.isi.edu.
                                A    255.255.255.0  ;aka X'FFFFFF00'

; Define one of the first level subnet numbers and masks
0.1.9.128.IN-ADDR.ARPA.         PTR  div1-subnet.isi.edu.
                                A    255.255.255.240 ;aka X'FFFFFFF0'

; Define another first level subnet number and mask
0.2.9.128.IN-ADDR.ARPA.         PTR  div2-subnet.isi.edu.
                                A    255.255.255.240 ;aka X'FFFFFFF0'

; Define second level subnet number
16.2.9.128.IN-ADDR.ARPA.        PTR  inc-subsubnet.isi.edu.
```

This assumes that the ISI network is named isi-net.isi.edu., first level subnets are named div1-subnet.isi.edu. and div2-subnet.isi.edu., and a second level subnet is called inc-subsubnet.isi.edu. (In a real system as complicated as this there would be more first and second level subnets defined, but we have shown enough to illustrate the ideas.)

4.3. Procedure for using an IP address to get network name

Depending on whether the IP address is class A, B, or C, mask off the high one, two, or three bytes, respectively. Reverse the octets, suffix IN-ADDR.ARPA, and do a PTR query.

For example, suppose the IP address is 10.0.0.51.

1. Since this is a class A address, use a mask x'FF000000' and get 10.0.0.0.

2. Construct the name 0.0.0.10.IN-ADDR.ARPA.

3. Do a PTR query. Get back

```
0.0.0.10.IN-ADDR.ARPA.  PTR     ARPANET.ARPA.
```

4. Conclude that the network name is "ARPANET.ARPA."

Suppose that the IP address is 128.9.2.17.

1. Since this is a class B address, use a mask of x'FFFF0000' and get 128.9.0.0.

2. Construct the name 0.0.9.128.IN-ADDR.ARPA.

3. Do a PTR query. Get back

```
0.0.9.128.IN-ADDR.ARPA.       PTR     isi-net.isi.edu
```

4. Conclude that the network name is "isi-net.isi.edu."

4.4. Procedure for finding all subnets involved with an IP address

This is a simple extension of the IP address to network name method. When the network entry is located, do a lookup for a possible A RR. If the A RR is found, look up the next level of subnet using the original IP address and the mask in the A RR. Repeat this procedure until no A RR is found.

For example, repeating the use of 128.9.2.17.

1. As before construct a query for 0.0.9.128.IN-ADDR.ARPA. Retrieve:

```
0.0.9.128.IN-ADDR.ARPA.  PTR    isi-net.isi.edu.
                         A      255.255.255.0
```

2. Since an A RR was found, repeat using mask from RR (255.255.255.0), constructing a query for 0.2.9.128.IN-ADDR.ARPA. Retrieve:

```
0.2.9.128.IN-ADDR.ARPA.  PTR    div2-subnet.isi.edu.
                         A      255.255.255.240
```

3. Since another A RR was found, repeat using mask 255.255.255.240 (x'FFFFFFF0'). constructing a query for 16.2.9.128.IN-ADDR.ARPA. Retrieve:

```
16.2.9.128.IN-ADDR.ARPA. PTR    inc-subsubnet.isi.edu.
```

4. Since no A RR is present at 16.2.9.128.IN-ADDR.ARPA., there are no more subnet levels.

5. YP ISSUES AND DISCUSSION

The term "Yellow Pages" is used in almost as many ways as the term "domain", so it is useful to define what is meant herein by YP. The general problem to be solved is to create a method for creating mappings from one kind of identifier to another, often with an inverse capability. The traditional methods are to search or use a precomputed index of some kind.

Searching is impractical when the search is too large, and precomputed indexes are possible only when it is possible to specify search criteria in advance, and pay for the resources necessary to build the index. For example, it is impractical to search the entire domain tree to find a particular address RR, so we build the IN-ADDR.ARPA YP. Similarly, we could never build an Internet-wide index of "hosts with a load average of less than 2" in less time than it would take for the data to change, so indexes are a useless approach for that problem.

Such a precomputed index is what we mean by YP, and we regard the IN-ADDR.ARPA domain as the first instance of a YP in the DNS. Although a single, centrally-managed YP for well-known values such as TCP-port is desirable, we regard organization-specific YPs for, say, locally defined TCP ports as a natural extension, as are combinations of YPs using search lists to merge the two.

In examining Internet Numbers [RFC 997] and Assigned Numbers [RFC 1010], it is clear that there are several mappings which might be of value. For example:

```
<assigned-network-name> <==> <IP-address>
<autonomous-system-id>  <==> <number>
<protocol-id>           <==> <number>
<port-id>               <==> <number>
<ethernet-type>         <==> <number>
<public-data-net>       <==> <IP-address>
```

Following the IN-ADDR example, the YP takes the form of a domain tree organized to optimize retrieval by search key and distribution via normal DNS rules. The name used as a key must include:

1. A well known origin. For example, IN-ADDR.ARPA is the current IP-address to host name YP.

2. A "from" data type. This identifies the input type of the mapping. This is necessary because we may be mapping something as anonymous as a number to any number of mnemonics, etc.

3. A "to" data type. Since we assume several symmetrical mnemonic <==> number mappings, this is also necessary.

This ordering reflects the natural scoping of control, and hence the order of the components in a domain name. Thus domain names would be of the form:

<from-value>.<to-data-type>.<from-data-type>.<YP-origin>

To make this work, we need to define well-know strings for each of these metavariables, as well as encoding rules for converting a <from-value> into a domain name. We might define:

```
<YP-origin>      :=YP
<from-data-type>:=TCP-port | IN-ADDR | Number |
                  Assigned-network-number | Name
<to-data-type>   :=<from-data-type>
```

Note that "YP" is NOT a valid country code under [ISO 3166] (although we may want to worry about the future), and the existence of a syntactically valid <to-data-type>.<from-data-type> pair does not imply that a meaningful mapping exists, or is even possible.

The encoding rules might be:

```
TCP-port            Six character alphanumeric

IN-ADDR             Reversed 4-octet decimal string

Number              decimal integer

Assigned-network-number
                    Reversed 4-octet decimal string

Name                Domain name
```

6. SPECIFICS FOR YP MAPPINGS

6.1. TCP-PORT

```
$origin Number.TCP-port.YP.

23                  PTR     TELNET.TCP-port.Number.YP.
25                  PTR     SMTP.TCP-port.Number.YP.

$origin TCP-port.Number.YP.

TELNET              PTR     23.Number.TCP-port.YP.
```

```
SMTP            PTR     25.Number.TCP-port.YP.
```

Thus the mapping between 23 and TELNET is represented by a pair of PTR RRs, one for each direction of the mapping.

6.2. Assigned networks

Network numbers are assigned by the NIC and reported in "Internet Numbers" RFCs. To create a YP, the NIC would set up two domains:

Name.Assigned-network-number.YP and Assigned-network-number.YP

The first would contain entries of the form:

```
$origin Name.Assigned-network-number.YP.

0.0.0.4         PTR     SATNET.Assigned-network-number.Name.YP.
0.0.0.10        PTR     ARPANET.Assigned-network-number.Name.YP.
```

The second would contain entries of the form:

```
$origin Assigned-network-number.Name.YP.

SATNET.         PTR     0.0.0.4.Name.Assigned-network-number.YP.
ARPANET.        PTR     0.0.0.10.Name.Assigned-network-number.YP.
```

These YPs are not in conflict with the network name support described in the first half of this RFC since they map between ASSIGNED network names and numbers, not those allocated by the organizations themselves. That is, they document the NIC's decisions about allocating network numbers but do not automatically track any renaming performed by the new owners.

As a practical matter, we might want to create both of these domains to enable users on the Internet to experiment with centrally maintained support as well as the distributed version, or might want to implement only the allocated number to name mapping and request organizations to convert their allocated network names to the network names described in the distributed model.

6.3. Operational improvements

We could imagine that all conversion routines using these YPs might be instructed to use "YP.<local-domain>" followed by "YP." as a search list. Thus, if the organization ISI.EDU wished to define locally meaningful TCP-PORT, it would define the domains:

<TCP-port.Number.YP.ISI.EDU> and <Number.TCP-port.YP.ISI.EDU>.

We could add another level of indirection in the YP lookup, defining the <to-data-type>.<from-data-type>.<YP-origin> nodes to point to the YP tree, rather than being the YP tree directly. This would enable entries of the form:

```
IN-ADDR.Netname.YP.    PTR      IN-ADDR.ARPA.
```

to splice in YPs from other origins or existing spaces.

Another possibility would be to shorten the RDATA section of the RRs which map back and forth by deleting the origin. This could be done either by allowing the domain name in the RDATA portion to not identify a real domain name, or by defining a new RR which used a simple text string rather than a domain name.

Thus, we might replace

```
$origin Assigned-network-number.Name.YP.

SATNET.          PTR     0.0.0.4.Name.Assigned-network-number.YP.
ARPANET.         PTR     0.0.0.10.Name.Assigned-network-number.YP.
```

with

```
$origin Assigned-network-number.Name.YP.

SATNET.          PTR     0.0.0.4.
ARPANET.         PTR     0.0.0.10.
```

or

```
$origin Assigned-network-number.Name.YP.

SATNET.          PTT     "0.0.0.4"
ARPANET.         PTT     "0.0.0.10"
```

where PTT is a new type whose RDATA section is a text string.

7. ACKNOWLEDGMENTS

Drew Perkins, Mark Lottor, and Rob Austein contributed several of the ideas in this RFC. Numerous contributions, criticisms, and compromises were produced in the IETF Domain working group and the NAMEDROPPERS mailing list.

8. REFERENCES

[HR] Braden, B., editor, "Requirements for Internet Hosts", RFC in preparation.

[ISO 3166] ISO, "Codes for the Representation of Names of Countries", 1981.

[RFC 882] Mockapetris, P., "Domain names - Concepts and Facilities", RFC 882, USC/Information Sciences Institute, November 1983.

Superseded by RFC 1034.

[RFC 883] Mockapetris, P.,"Domain names - Implementation and Specification", RFC 883, USC/Information Sciences Institute, November 1983.

Superceeded by RFC 1035.

[RFC 920] Postel, J. and J. Reynolds, "Domain Requirements", RFC 920, October 1984.

Explains the naming scheme for top level domains.

[RFC 952] Harrenstien, K., M. Stahl, and E. Feinler, "DoD Internet Host Table Specification", RFC 952, SRI, October 1985.

Specifies the format of HOSTS.TXT, the host/address table replaced by the DNS

[RFC 973] Mockapetris, P., "Domain System Changes and Observations", RFC 973, USC/Information Sciences Institute, January 1986.

Describes changes to RFCs 882 and 883 and reasons for them.

[RFC 974] Partridge, C., "Mail routing and the domain system", RFC 974, CSNET CIC BBN Labs, January 1986.

Describes the transition from HOSTS.TXT based mail addressing to the more powerful MX system used with the domain system.

[RFC 997] Reynolds, J., and J. Postel, "Internet Numbers", RFC 997, USC/Information Sciences Institute, March 1987

Contains network numbers, autonomous system numbers, etc.

[RFC 1010] Reynolds, J., and J. Postel, "Assigned Numbers", RFC 1010, USC/Information Sciences Institute, May 1987

Contains socket numbers and mnemonics for host names, operating systems, etc.

[RFC 1034] Mockapetris, P., "Domain names - Concepts and Facilities", RFC 1034, USC/Information Sciences Institute, November 1987.

Introduction/overview of the DNS.

[RFC 1035] Mockapetris, P., "Domain names - Implementation and Specification", RFC 1035, USC/Information Sciences Institute, November 1987.

DNS implementation instructions.

Author's Address:

Paul Mockapetris
USC/Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292

Phone: (213) 822-1511

Email: PVM@ISI.EDU